CCured Manual

Installation:

1. Make sure your base system is Ubuntu 16.04, this tool hasn't been tested on other platforms up to now, so Ubuntu 16.04 is the preferred choice.

2. Download **LLVM 6.0.0** source code package from the link below: https://releases.llvm.org/download.html

3. Copy the attachment "NesCheck" directory into "**/where/is/your/llvm-6.0.0-src/lib/Transforms**" directory.

4. Modify the CMakeLists.txt in the LLVM 6.0 source code root directory, add "**add_subdirectory(lib/Transforms/NesCheck)**" at line 849.

5. Download and install CMake, version 3.4.3 is the minimum required, recommend using the command **apt-get install cmake** to install it rather than installing from source code.

6. Create a build directory for building LLVM 6.0, since LLVM 6.0 doesn't support for building in the same directory. Create it using command **mkdir LLVMbuild**. Make sure **LLVMbuild** is **NOT** inside the LLVM **source** directory

7. Enter the build directory using **cd LLVMbuild**, then execute the command below: **cmake /where/is/your/llvm-6.0.0-src/**.

8. Start building LLVM 6.0 in the LLVMbuild directory using command below: **cmake --build .** (Don't forget there is a **.** in the command).

9. If everything goes well, you should be able to see a **Makefile** in the **LLVMbuild/** directory, run **make** to build LLVM 6.0 (takes time).

Note: Step 5-9 is for building LLVM 6.0 using **CMake**, you can also refer to the link https://releases.llvm.org/6.0.0/docs/CMake.html for further details.

10. After building LLVM 6.0, check whether it contains the library file **libNesCheck.so** in the **LLVMbuild/lib** directory, if it is there, then the NesCheck pass is successfully installed.

Please refer to the next page for usage.

Usage:

1. Extracting bitcode for the being analyzed program using **wllvm**, here is the link for installing and using wllvm: https://github.com/travitch/whole-program-llvm. Make sure the **Clang** version is **6.0** when you use **wllvm**. Note that there should be multiple ways to extract bitcode from binaries, **wllvm** is just one of them.

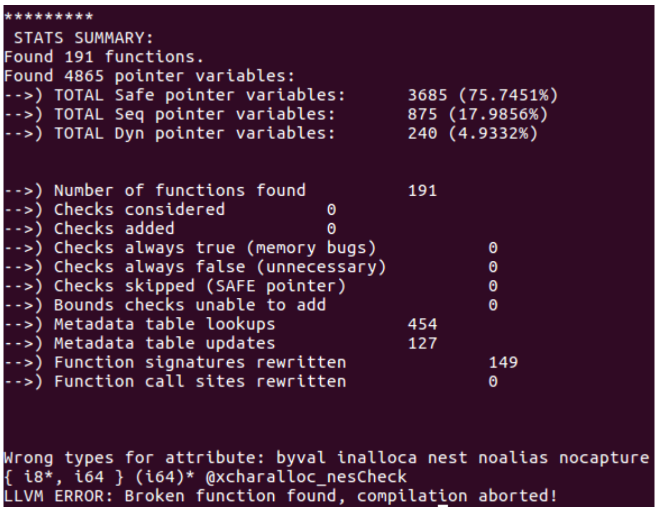
2. Generating bitcode for the attached neschecklib.c using command below:
   clang-6.0 -O0 -g -emit-llvm neschecklib.c -c -o neschecklib.bc

3. Link the bitcode generated in Step 1 to the neschecklib.bc using command below:
   LLVMbuild/bin/llvm-link neschecklib.bc target.bc -o target.linked.bc

Note: Step 3 is **REALLY** important, it prevents null pointers while analyzing programs.

4. Run CCured analysis using the command below:
   LLVMbuild/bin/opt -o target.opt.bc -load LLVMbuild/lib/libNesCheck.so -nescheck -stats -time-passes < target.linked.bc > target.nescheckout

5. Result will be shown in command line once the analysis finished: Here is the sample example for analyzing **chmod** in **coreutils:**



```
*********
 STATS SUMMARY:
Found 191 functions.
Found 4865 pointer variables:
-->) TOTAL Safe pointer variables:      3685 (75.7451%)
-->) TOTAL Seq pointer variables:       875 (17.9856%)
-->) TOTAL Dyn pointer variables:       240 (4.9332%)


-->) Number of functions found          191
-->) Checks considered          0
-->) Checks added               0
-->) Checks always true (memory bugs)           0
-->) Checks always false (unnecessary)          0
-->) Checks skipped (SAFE pointer)              0
-->) Bounds checks unable to add                0
-->) Metadata table lookups             454
-->) Metadata table updates             127
-->) Function signatures rewritten              149
-->) Function call sites rewritten              0



Wrong types for attribute: byval inalloca nest noalias nocapture
{ i8*, i64 } (i64)* @xcharalloc_nesCheck
LLVM ERROR: Broken function found, compilation aborted!
```

Note that there may be an LLVM ERROR shows in command line once analysis is finished, this is caused by function rewriting, and it corrupted the LLVM's own output of statistics. Since the error happens when analysis is finished and the statistics can also be output by the pass itself, the error isn't an issue and won't have any influence towards the analysis.